Distributed Environments for Large Data-Objects[1]: Broadband Networks and a New View of High Performance, Large-Scale Storage-Based Applications

William Johnston, Jin Guojun, Gary Hoo, Case Larsen
Jason Lee, Brian Tierney, Mary Thompson
Imaging and Distributed Computing Group
Ernest Orlando Lawrence Berkeley National Laboratory
University of California
Berkeley, CA, 94720

Abstract

We are exploring the use of highly distributed architectures in wide area, broadband networks for all aspects of collecting, storing, analyzing, and otherwise manipulating and making generally available, large data-objects. These objects - typically the result of a single operational cycle of a scientific or medical instrument, and of sizes from tens of MBytes to tens of Gbytes - are the staple of modern analytical systems. In this paper, we describe the network storage system architecture, client access mechanisms, and security, and the implications for these aspects while operating in a wide area network. We specifically describe the use of a shared (public) IP over ATM network to facilitate collection, storage, analysis, distribution, and delivery of several kinds of large data-objects.

1.0 Introduction

The advent of shared, widely available, high-speed networks is providing the potential for new approaches to the collection, storage, and analysis of large data-objects. In one example, high-volume health care image data used for diagnostic purposes - e.g. X-ray CT, MRI, and cardio-angiography - are increasingly collected at tertiary (centralized) facilities, and may now be routinely stored and used at locations other than the point of collection. The importance of distributed storage is that a hospital (or any other instrumentation scenario) may not be provide best environment in which to maintain a large-scale digital storage system, and an affordable, easily accessible, high-bandwidth network can provide location independence for such storage. In this scenario, the importance of remote end-user access is that the health care professionals at the referring facility (frequently remote from the tertiary imaging facility) will have ready access to not only the image analyst's reports, but the original image data itself.

This general scenario extends to other fields as well. In particular, the same basic infrastructure is required for remote access to large-scale scientific and analytical instruments, both for data handling and for direct, remote-user operation. See [Johnston95b].

The basic elements of the overall large data-object architecture include:

- data collection and the instrument-network interface
- on-line storage that is distributed throughout the network (for both performance and reliability)
- processing elements - also distributed throughout the network - for various sorts of data analysis
- data management that provides for the automatic cataloguing (metadata generation) of the data being stored
- data access interfaces, including application-data interfaces
- tertiary storage (“mass storage”) management
- user access to all relevant aspects (application, data, metadata, data management)
- transparent security that provides access control for all of the systems components based on the resource-owner's policy

Within the network storage system in particular - a “middleware” service - the architectural issues include:

- distributed storage system operation and performance
- user access methodologies
- security architecture

These elements all need to be provided with flexible, location-independent interfaces so that they can be freely moved around the network as required for operational or other logistical convenience.

1. This work is supported by the U. S. Dept. of Energy, Energy Research Division, Mathematical, Information, and Computational Sciences office (http://www.er.doe.gov/production/octr/mics), under contract DE-AC03-76SF00098 with the University of California, and by ARPA ITO (http://ftp.arpa.mil/ResearchAreas.html). Author's address: Mail Stop: 50B-2239. Tel: +1-510-486-5014, fax: +1-510-486-6363, wejohnston@lbl.gov, http://www-itg.lbl.gov/~johnston. This document is report LBNL-38969.

2.0 Network-Based Storage: The Distributed-Parallel Storage System

Widely distributing the components of a storage system throughout networks increases its capacity, reliability, performance, and security. Usable capacity increases in conjunction with a widely deployed, generalized security infrastructure that can support mechanisms for dynamic construction of systems through brokering and automated acquisition of resources. Reliability increases because storage systems that can be configured from components that have as little as possible in common (e.g. location) provide the resilience that comes from independence. Performance is increased by the combined characteristics of parallel operation of many sub-components, and the independent data paths provided by a large network infrastructure. Security is also potentially increased by having many independent components each of which has local and independent enforcement mechanisms that can limit the scope of a security breach.

The Distributed-Parallel Storage System (DPSS, also known as ISS) is an experimental system in which we are developing, implementing, and testing these ideas. In most configurations, it is used as a network-striped disk array designed to supply and consume high-speed data streams to and from other processes in the network. (See [DPSS].)

2.1 DPSS Architecture

The DPSS is essentially a "logical block" server whose functional components are distributed across a wide-area network. (See Figure 1 illustrating the DPSS architecture.) The DPSS uses parallel operation of distributed servers to supply data

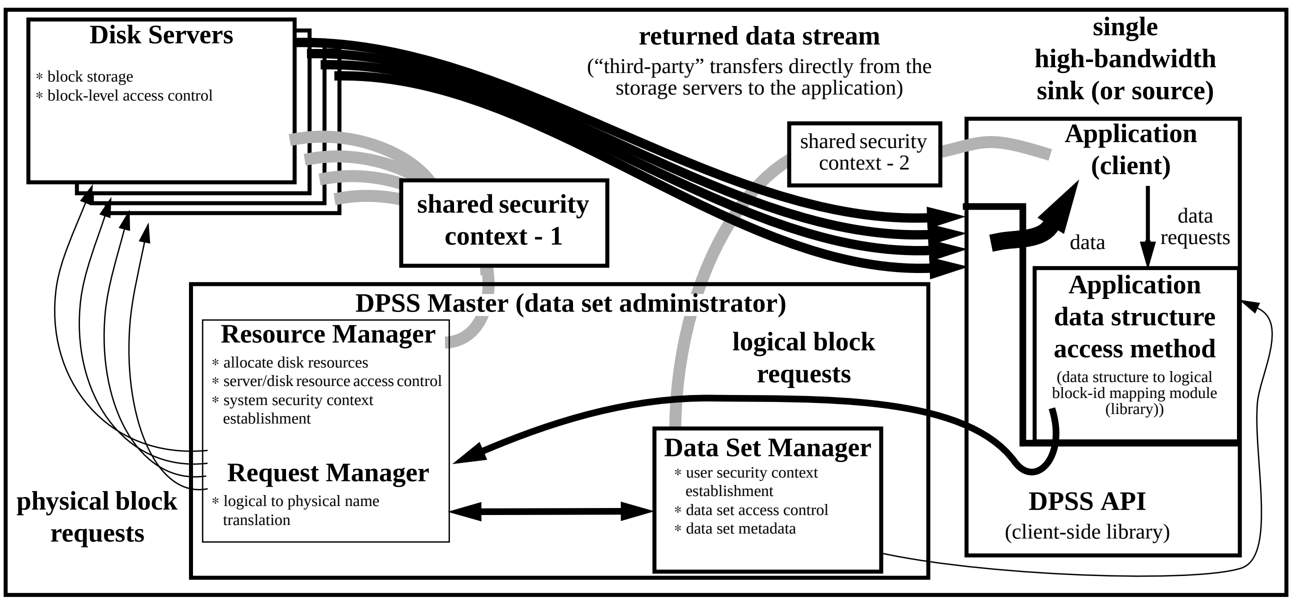

Figure 1 **Distributed-Parallel Storage System Architecture**

streams fast enough to enable various multi-user, "real-time", virtual reality-like applications in an Internet / ATM environment. There is no inherent, DPSS specific organization to the data blocks, and in particular they would never be organized sequentially on a server. The data organization is determined by the application as a function of data type and access patterns, and is implemented during the data load process. The usual goal of the data organization is that data is declustered (dispersed) across both disks and servers. This strategy allows a large collection of disks to seek in parallel, and all servers to send the resulting data to the application in parallel, enabling the DPSS to perform as a high-speed data server.

The implementation is based on the use of multiple low-cost, medium-speed disk servers that use the network to aggregate multiple server outputs for high performance applications. To achieve high performance all types of parallelism are exploited, including those available at the level of the disks, controllers, processors / memory banks, servers, and the network.

At the application level, the DPSS is a semi-persistent cache of named objects, and at the storage level it is a logical block server. Although not strictly part of the DPSS architecture, the system is usually used with an application agent called a "data set structure method". This component provides an object-like encapsulation of complex user-level data structures so that the application does not have to retain this information for each different data set. The structure server converts the application data requests into logical block requests. These logical block requests are then sent to the DPSS "data set administrator" (DSA). The DSA maintains data set definitions, and is responsible for mapping the logical block requests to physical block requests. The DSA also deals with interactions with the storage servers to determine available storage (a storage server is an independent entity and may deal with several DPSS masters) and to establish the “security context” that provides the scope of control for various resources.

The security model for the DPSS involves accommodating several different resource owners. The context established between the DSA and the disk/storage servers reflects an agreement between the owner of the physical resources (disks) and an agent that is providing storage to a user community. This context enforces the disk usage agreement. A separate context established between the DSA and the users reflects the use-conditions imposed by the data “owner”, and provides for ensuring access control that enforces those use-conditions on data.

The overall data flow involves “third-party” transfers from the storage servers directly to the data-consuming application (a model used by most high performance storage systems). Thus, the application requests data, these requests are eventually translated to physical block addresses (server name, disk, and disk block), and the servers deliver data directly to the application. (The servers and the application host are made known to each other during the data set “open” process, and sufficient information from the original request is carried along so that the application can identify the data provided by the servers.)

The DPSS is not an inherently reliable tertiary storage system, though the potential exists for this capability by adding tertiary backing storage on the storage servers.

3.0 Applications

TerraVision - Large-Image Browsing: The initial use of the DPSS was to provide data to a terrain visualization application in the MAGIC testbed [MAGIC]. This application, known as TerraVision, allows a user to navigate through and over a high resolution landscape represented by digital aerial images and elevation models [TerraVision]. TerraVision is very different from a typical “flight simulator”-like program in that it uses high-resolution aerial imagery for the visualization instead of simulated terrain. TerraVision requires large amounts of data, transferred at both bursty and steady rates. The DPSS is used to supply image data at hundreds of Mbits/s to TerraVision.

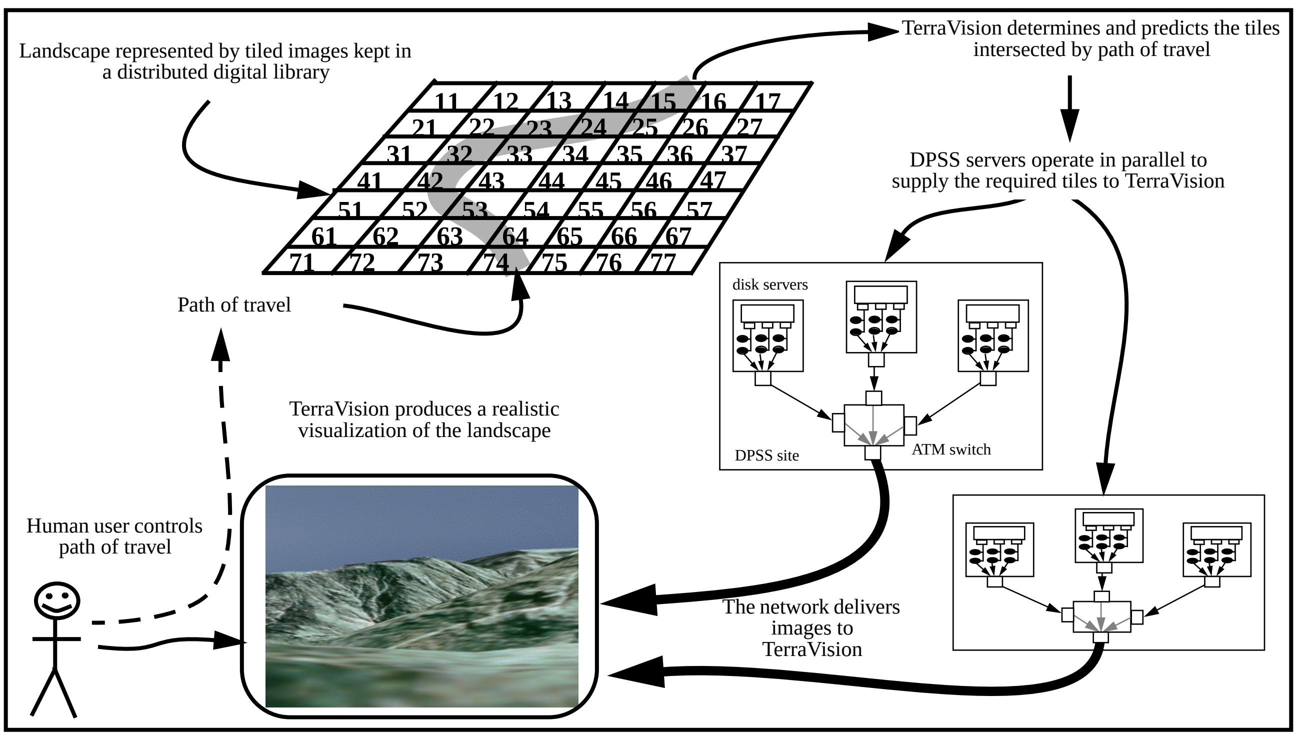

Figure 2 **TerraVision and the DPSS Cooperate to Visualize Landscape**

In the case of a large-image browsing application like TerraVision, the strategy for using the DPSS is straightforward: the image is tiled (broken into smaller, equal-sized pieces), and the tiles are scattered across the disks and servers of the ISS. The order of tiles delivered to the application is determined by the application predicting a “path” through the image (landscape) and requesting the tiles needed to supply a view along the path. (See Figure 2.) The actual delivery order is a function of how quickly a given server can read the tiles from disk and send them over the network. Tiles will be delivered in roughly the requested order, but variations from the requested order will occur. These variations must be accommodated by buffering, or other strategies, in the client application. The predictive aspect of the operation is important because the DPSS is heavily pipelined, and the pipeline must be kept full in order to obtain good performance.

This approach can supply data to any sort of large-image browsing application, including applications for displaying large aerial-photo landscapes, satellite images, X-ray images, scanning microscope images, and so forth.

Figure 3 illustrates the configuration and operation of the DPSS in the ARPA-funded MAGIC high-speed ATM testbed

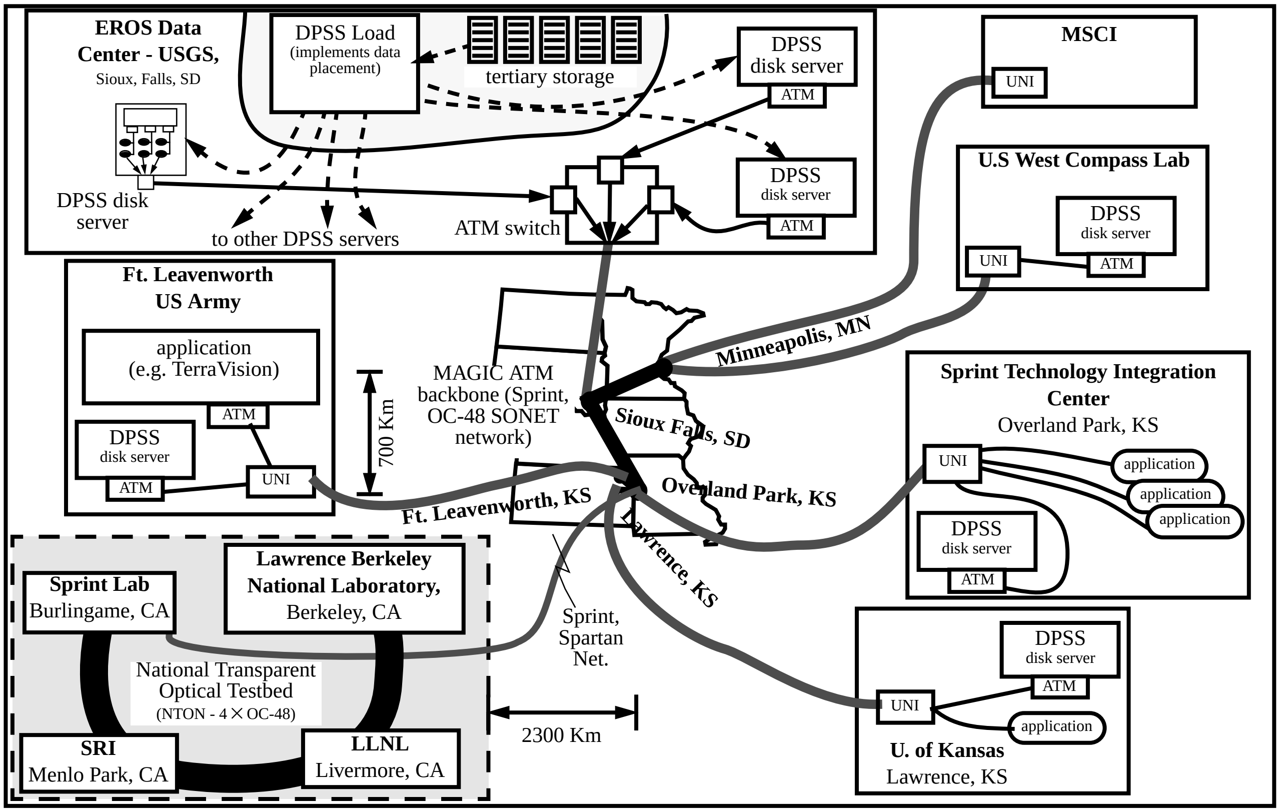

Figure 3 **The MAGIC Testbed Distributed Application Environment**

where the DPSS is being developed. In the MAGIC testbed, the DPSS has been run in numerous ATM WAN configurations to drive several different applications, including TerraVision. The configurations include placing DPSS servers in Sioux Falls, South Dakota, Overland Park, Kansas, and Lawrence, Kansas, and running the TerraVision client at Fort Leavenworth, Kansas, and at SRI in California. In MAGIC the DPSS disk servers and the TerraVision application are separated by several hundred kilometers, the longest single link being about 700 kilometers. With the application running at SRI, the longest link is about 2300 Km.

A Health Care Image Management System: An example of a medical application that uses a distributed large data-object architecture is a system that provides for collection, storage, cataloguing, and playback of video-angiography[2] images using a shared metropolitan area ATM network.

The image data are sent through the network to storage and analysis systems, as well as directly to users at clinic sites. Thus, data can be stored and catalogued for later use, data can be delivered live from the imaging device to remote clinics in real-time, or these data flows can all be done simultaneously. Whether the storage servers are local or distributed around the network is entirely a function of the optimal logistics: There are arguments in regional health care information systems for centralized storage facilities away from the hospital environment, even though the architecture is that of a distributed system. (See [Johnston93].)

Figure 4 illustrates the configuration of the health care application as it is embedded in a Pacific Bell ATM network. This application is a joint project of Lawrence Berkeley National Laboratory, Kaiser Permanente, Philips Research (Palo Alto),

2. Cardio-angiography imaging involves a two plane, X-ray video imaging system that produces from several to tens of minutes of digital video sequences for each patient study for each patient session. The digital video is organized as tens of data-objects, each of which are of the order of 100 MBytes.

and the Pacific Bell CalREN program. (See [Kaiser-CalREN].) The network aspects of this health care project are closely

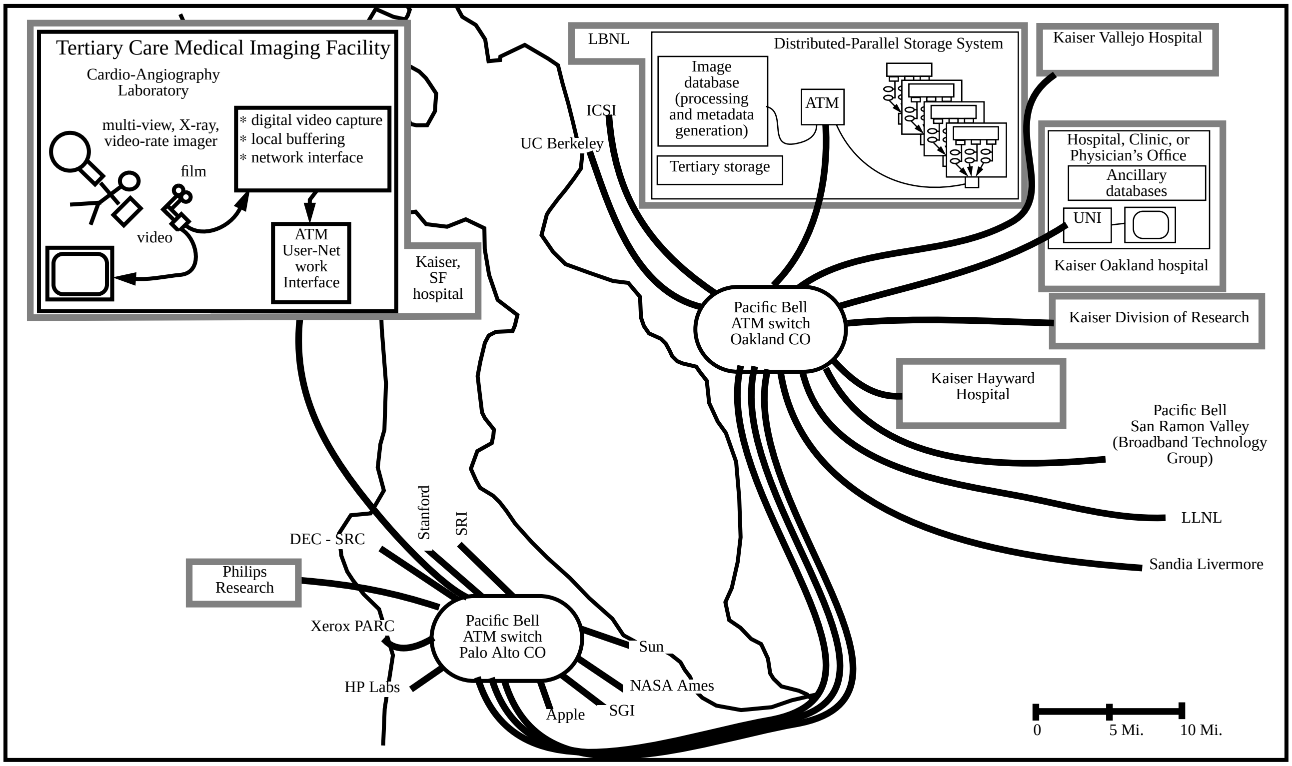

Figure 4 Distributed Large Data-Object Medical Imaging Application Embedded in the Pacific Bell, S. F. Bay Area, Metropolitan Area ATM Network (BAGNet and Kaiser CalREN Sites Shown)

related to the BAGNet CalREN project (see [BAGNet-1] and [Wiltzius]). The the logical and network infrastructure are described in [Johnston95c].

The angiography data is being collected directly from a Philips scanner by a computer system at the San Francisco Kaiser Cardiac Catheterization Laboratory. This computing system, in turn, is attached to the metropolitan area ATM network. When the data collection for a patient is complete (about once every 20-40 minutes), about 500 MBy of digital video data is sent across the network to LBNL (in Berkeley) and stored first on the DPSS, and then is archived to a mass storage system. This process goes on 8-10 hours a day.

The data is catalogued in a World Wide Web based image database system ("ImgLib"). Use of the data involves access through ImgLib to locate the data sets of interest, and then to invoke a viewing application. Department-level Web-based patient databases can refer directly to the data in ImgLib without duplicating the data, or being concerned about tertiary storage management (which is handled by ImgLib).

4.0 Overall Large Data-Object System Architecture

4.1 Data Management

In any scenario where data is generated in large volumes and with high throughput, and especially in a distributed environment where the people generating the data and the people cataloguing and using the data are geographically separated, there are several important issues: automatic generation of at least minimal metadata; cataloguing of the data and the metadata as the data is received (or as close to real time as possible); transparent management of multiple tertiary storage systems; and facilitation of co-operative research by allowing specified users at local and remote sites immediate access to the data, and thought incorporation of the data into other databases or documents.

Our approach to this capability (called "ImgLib") is to use the World Wide Web tools to provide this library-like functionality. Semi-automatic cataloguing of incoming data is done by extracting associated metadata and converting it into text records, by generating auxiliary metadata and derived data, and by combining these into Web documents. (See [ImgLib].) Tertiary storage management is provided by using the remote program execution capability of Web servers to provide transparent access to different kinds of mass storage systems that then return the data-objects to the Web server, or, as is the case in several of our applications, move the data to a DPSS cache for access by applications. (For an example of a Java applet accessing the DPSS, see http://www-itg.lbl.gov/ISS/browser/iss2d.html)

Figure 5 illustrates some of these points. It shows a browser/query interface on the right, and on the left the results of

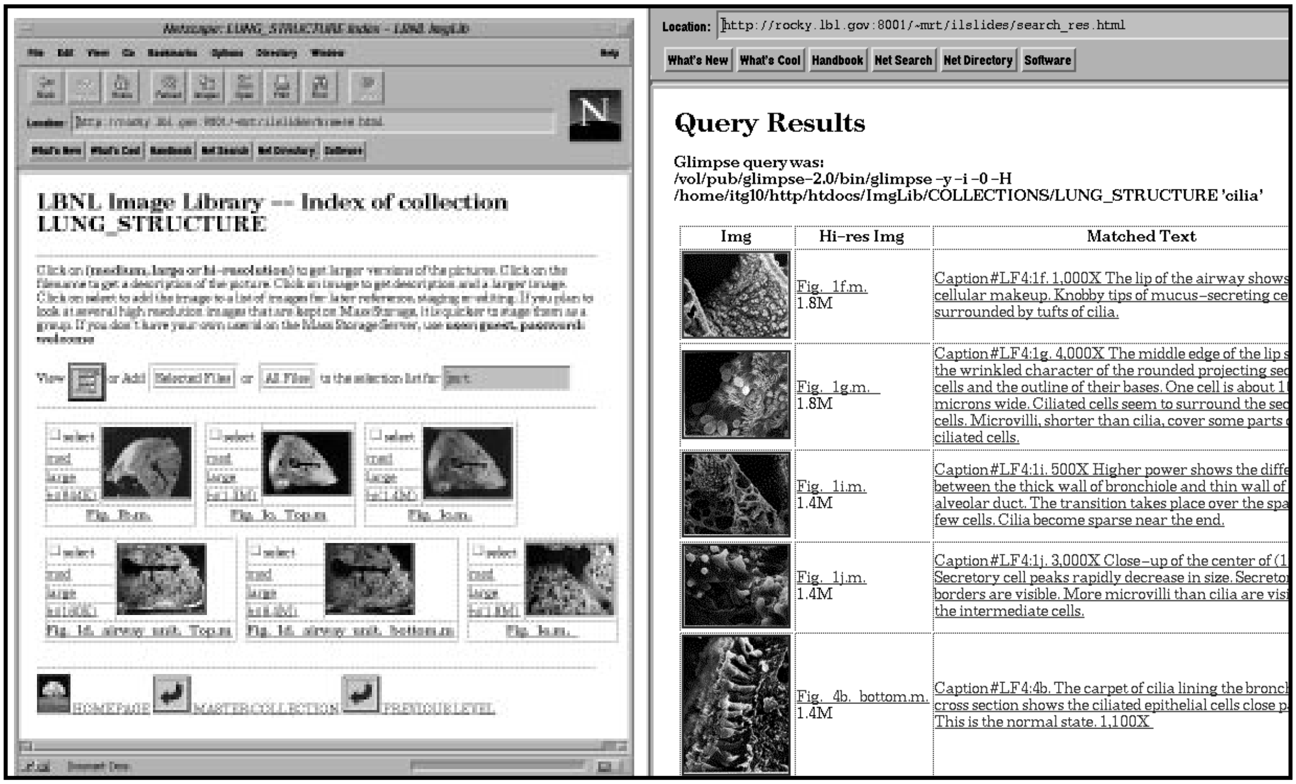

Figure 5 **Large Data-Object Management for Video-like Objects**

automatically building a sub-collection as the result of a search on the textual metadata. The information about the data-objects that result from the search is shown as an associated collection of thumbnails, associated pointers to several other types of derived data (including a “movie” representation), and a pointer to the original data-object. For the data-objects that reside on tertiary storage (a tape-robot based mass storage system in this case), and there is an option for forcing migration of data back to the on-line cache if the data of interest is not already there. In this example, a JPEG “movie” is one of the derived representations. (The original video data cannot be compressed and requires a special application to view.)

5.0 Network Storage System Issues

5.1 Performance

Network performance and characteristics are obviously an important factor in the architecture that we are describing. There are virtually no behavioral aspects of an ATM “network” that can be taken for granted, even in an end-to-end ATM network. By “network” we mean the end-to-end data path from the transport API through the host network protocol (TCP/IP) software, the host network adaptors and their device drivers, the many different kinds of ATM switches and physical link bandwidths, and then up through the corresponding software stack on the receiver. Further, the behavior of different elements at similar places in the network architecture can be quite different because they are implemented in different ways. The combination of these aspects can lead to complex and unpredictable network behavior.

We have studied the behavior of TCP over both uncongested and congested AMT networks. These studies are described in “Performance Analysis in High-Speed Wide-Area ATM Networks: Top-to-Bottom End-to-End Monitoring” ([Tierney96]), and in a companion of this current paper (“Bay Area Gigabit Testbed Network (BAGNet): Experience with a Highspeed Metropolitan Area IP over ATM Networks”).

5.2 Latency

One of the primary issues for our approach of using the network as the backbone of a parallel storage system is the overall latency in the system. The whole system is heavily pipelined in order to achieve high performance. 100 Mbits/s of data per storage server using modern workstations as the platform is not difficult to achieve, leading to the capability of delivering many hundreds of Mbits/s to a single application. However, to achieve this sort of performance the storage servers must be able to keep all of the resources (mainly disks and the network interface) operating continuously at full capacity. This, in turn, requires that the application be able to predict ahead the “length of the pipeline”. (Which is about 150-175 ms.)

This sort of prediction is easy for some classes of applications (e.g. a video server, and even an interactive multimedia server - see [Tierney94]), is fairly difficult for some applications (TerraVision uses a “path constraint” algorithm to predict what parts of the image data will be needed in the near future, and deliberately requests more data than it will actually use in order to ensure that the required data is available when needed for the realtime visualization), and would be impossible for a random access application (though these are unusual, and no storage system works well under these circumstances).

Because of the importance of latency by have done both component-level studies (Figure 6) and empirical studies (Figure 7). Figure 7 shows the results of measurements in both LAN and (uncongested) WAN environments

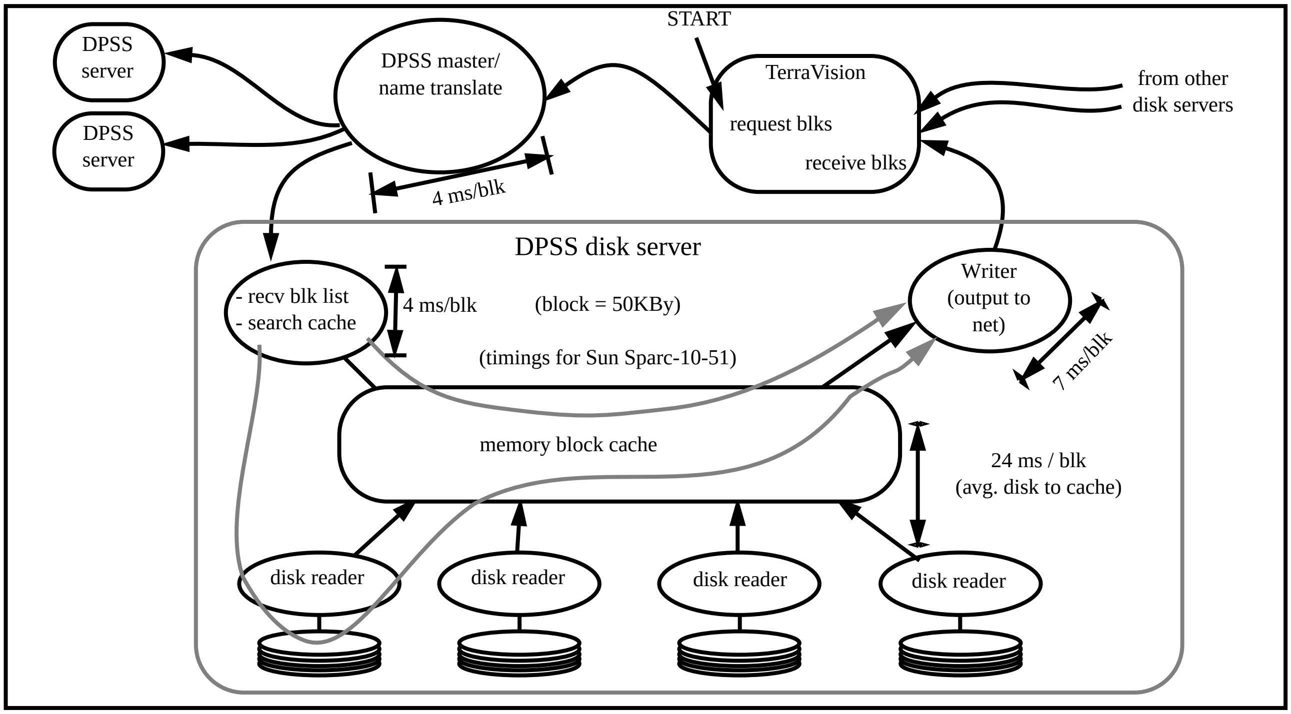

Figure 6 **DPSS Latency Characterization**

(“MAGIC”) and represents the cumulative time through a system (from request to receipt of data), with time measurements made at the points indicated in Figure 6.

6.0 Security Architectures for Large-Scale Remote Environments

The access to remote resources via open networks, whether for monitoring, control of experiments, or manipulation of data, requires that authentication and access control mechanisms be in place in order to provide safeguards against unauthorized access to resources, and to assure privacy of proprietary, confidential, or otherwise restricted data. Therefore, awkward, intrusive, and expensive security will delay wide-spread use of remote resources in open environments.

In addition, we would like the security architecture and supporting infrastructure to be general enough to enable automatic brokering of resources. This new capability could support a scenario where resource owners (either as a mainline business, or as a barter-based use of excess capacity) advertise resources and user agents collect and assemble these resources into useful systems. (This could be a very useful service in the world of scientific experiment control, when significant resources may be required only for well-defined periods of time, and idle otherwise.) It is our hypothesis that this capability can be built on the same use-condition based security architecture that is described here.

In order to provide the required capabilities in a manageable, cost effective, and usable manner the security architecture must be flexible and effective, and easily deployed, administered, and used. In order to realize its potential not only for transparent access control, but also as the basis of an automated resource brokering system, all aspects of the security process after certificates are obtained must be automated. In order that the approach be scalable, the components and operating aspects should be decentralized.

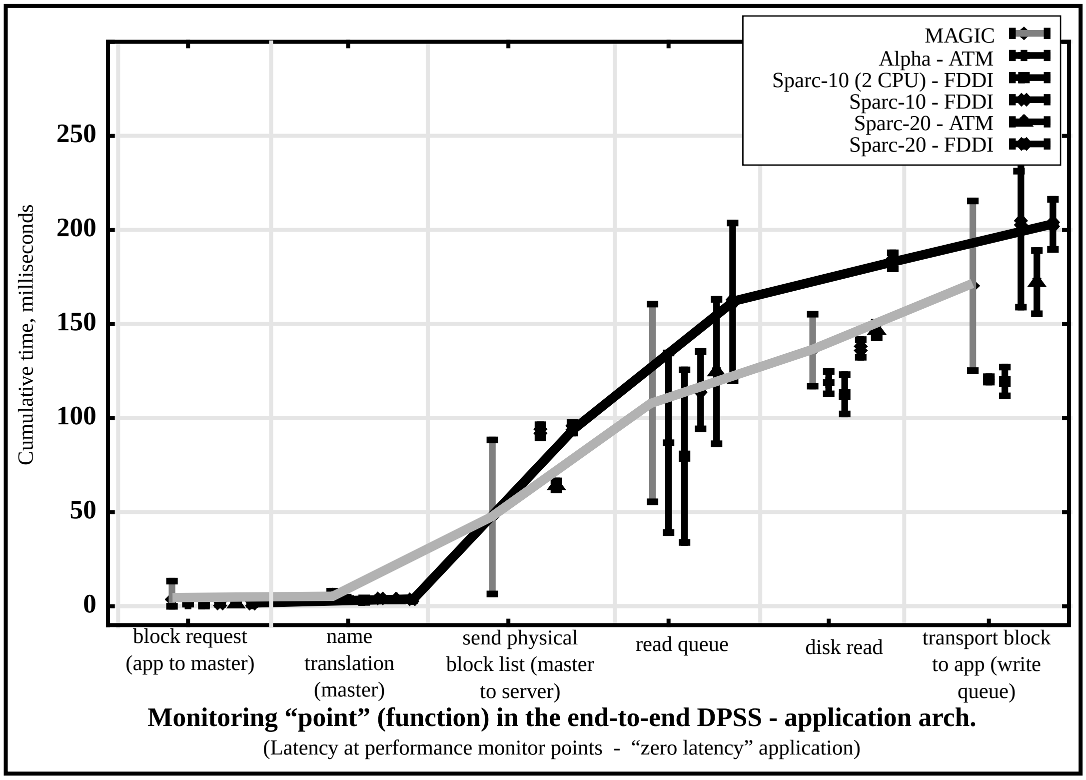

Figure 7

DPSS "Open Loop" Performance: Latencies in the Architecture

The problem, then, is to identify and demonstrate authentication and access control methodologies that are applicable to all user-level resources - data, storage and processing, control of external devices, etc. - in widely distributed systems, and that are also general, scalable, "strong", non-intrusive (for both the administrator and the user), and easily managed.

The overall goal of the security architecture is to be able to encode, distribute, protect, and then act on, information that is needed to provide routine, secure availability of remote resources in a widely distributed environment, and to enable a level of automated processing that will permit resource brokering.

Background: Public-key certificates (PKC) are an inherently reliable and secure mechanism for widely distributing the assured information needed for authentication and authorization. (See [RSA] for background information.) Public-key certificates can encode information about a principal, or information expressed by a principal, or a relationship between principals, in a secure and verifiable way. Certificates that provide some policy based assurances of the identity of the principal we call identity certificates. Certificates that encode organizational / group affiliation, creditworthiness, level and scope of training, etc., we call "credentials". Certificates that encode use-conditions, e.g., cost, required role attributes (personal identity, group membership, organizational function), required personal characteristics (training, credit worthiness, etc.) we call requirement certificates. Certificates that encode a relationship of a principle to a policy (e.g. that one certification authority operates under the policy of another) we call trust certificates.

6.1 A Use-Condition Centered Security Model

We are developing a use-condition centered model that is based on verifiable and securely represented resource controller / owner imposed use-conditions, and equally secure and verifiable satisfaction of those conditions. The individual identity that is so important in many security architectures is just one of many factors in our model.

The basic idea is that the principals that control resources will establish a set of conditions for the use of the resources. The use-conditions are whatever is appropriate to the resource: a level of training for operating an instrument; membership in a group (e.g. health care professionals working for a particular HMO) for access to patient data; creditworthiness for access to a resource that requires a commitment of payment; etc. Use-conditions are encoded in signed certificates and deposited in a "public" server.

Paired with the resource owner's use-conditions are the principals that can attest to the relevant attributes of a user or agent that is seeking access to a resource: A school registrar may certify a level of training; a state examining board may certify a level of professional attainment; an organization can certify employment (and related group membership); a bank, a level of credit; and a certification authority may attest to individual identity. Again, the principal (person or organization) that can attest to some attribute, expresses that fact in a signed certificate that is deposited in a “public” server.

A capability to use a resource is represented by a composite certificate that contains all of the use-conditions and certificates indicating their satisfaction. This composite certificate is most likely formed by a mutually trusted third party (agent) on behalf of the resource owner and the user.

The last part of the model are “deferred” verifications. These actions may be required in the use-conditions, but are deferred to the resource access control mechanism, or may be imposed at the resource as operational policy. The consuming of a “cybercash” unit of the resource is an example of this. Other examples include checking all manner of time sensitive requirements, re-verification of user credentials in the case of high-consequence access, etc.

For more information on the security model please see “Security Architectures for Large-Scale Remote Collaboratory Environments: A Use-Condition Centered Approach to Authenticated Global Capabilities” ([Johnston96a]).

6.2 Implementation of a Prototype Architecture

Our initial prototype is illustrated in Figure 8, and it addresses:

- User authentication, credential, and key management via SSH [SSH]
- Identity certificate management via LDAP/X.500 [LDAP] / WWW
- Resource access control
  - security context establishment via GSS/SPKM [Adams]
  - secure, authenticated, and integrity protected exchanges via GSS-API [GSS]

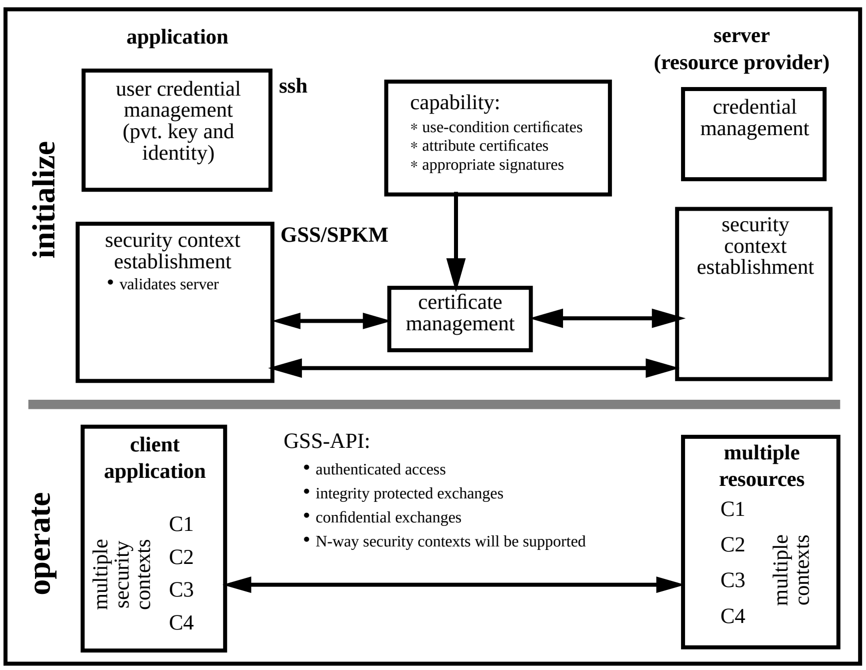

Figure 8

Current Prototype Security Architecture Implementation

6.3 Prototype Application

We are using the prototype implementation to provide security to an experimental health care information application (see [Kaiser-CalREN]) that uses the Distributed-Parallel Storage System. The application uses a WWW-based system for managing certain kinds of medical imaging records. The data-object viewing application (“Vplayer”) is invoked via a WWW image database frontend that uses WWW security to protect the metadata. The application (a specialized, combined video and image browser) then accesses data stored on the DPSS. The Vplayer access to DPSS data is protected by the security architecture described here. (That is, credentialing is handled by the secure shell that runs Vplayer, and the DPSS access is controlled by checking authorization certificates presented by Vplayer.)

7.0 Conclusions

We have described several elements of an architecture the using high-speed, wide-area networks as the basis of a highly distributed environment for handling all aspects of large data-objects. Variations of the overall approach have been applied in several different projects, and we have described some of the results, in particular as they illustrate network-based storage systems.

We believe this approach to be both practical and powerful given the increasing availability of high-speed networks. Further work will be done on both the various functional elements and specific applications. Please see http://www-itg.lbl.gov for more information.

8.0 Notes and References

Adams Adams, C. “IDUP and SPKM: Developing Public-Key-Based APIs and Mechanisms for Communication Security Services”. See http://bilbo.isu.edu/sndss/sndss96.html

BAGNet-1 “Bay Area Gigabit Testbed (BAGNet)” See http://www-itg.lbl.gov/BAGNet.html

DPSS The Distributed-Parallel Storage System (DPSS) Home Page (http://www-itg.lbl.gov/DPSS)

GSS Linn, J., “Generic Security Service Application Program Interface” (http://ds.internic.net/rfc/rfc1508.txt) Also see more recent and related drafts at the IETF Common Authentication Technology home page (http://www.ietf.cnri.reston.va.us/html.charters/cat-charter.html).

ImgLib See http://www-itg.lbl.gov/ImgLib/ImgLib_intro.html

Johnston96a Johnston, W. and C. Larsen, “Security Architectures for Large-Scale Remote Collaboratory Environments: A Use-Condition Centered Approach to Authenticated Global Capabilities” (draft at http://www-itg.lbl.gov/~johnston)

Johnston95b W. Johnston, and D. Agarwal, “The Virtual Laboratory: Using Networks to Enable Widely Distributed Collaboratory Science” A NSF Workshop Virtual Laboratory whitepaper. (See http://www-itg.lbl.gov/~johnston/Virtual.Labs.html)

Johnston95c Johnston, W., “BAGNet: A High-Speed, Metropolitan Area, IP over ATM Network Testbed”, IEEE Compcon’95, March, 1995, San Francisco, CA. Also see http://www-itg.lbl.gov/BAGNet.hm.pg.docs/CompCon.95.3.html .

Johnston93 W. Johnston and A. Allen, “Regional Health Care Information Systems: Motivation, Architecture, and Implementation,” LBL Technical Report 34770 (draft), Dec. 1993.

Kaiser-CalREN Kaiser - LBNL - Philips CalREN project. See http://www-itg.lbl.gov/Kaiser/LKP/homepage.html .

LDAP “Lightweight Directory Access Protocol”, http://www.umich.edu/~rsug/ldap/ldap.html.

MAGIC MAGIC (Multidimensional Applications and Gigabit Internetwork Consortium) is a gigabit network testbed that was established in June 1992 by the U. S. Government’s Advanced Research Projects Agency (ARPA). The testbed is a collaboration between LBNL, Minnesota Supercomputer Center, SRI, Univ. of Kansas, Lawrence, KS, USGS - EROS Data Center, CNRI, Sprint, U. S. West, Southwest Bell, and Splitrock Telecom. More information about MAGIC may be found on the WWW home page at: http://www.magic.net/

RSA “RSA Labs’ Frequently Asked Questions About Todays’ Cryptography” http://www.rsa.com/rsalabs/faq/faq_home.html

SSH Ylonen, T., “The SSH (Secure Shell) Remote Login Protocol” (ftp://ietf.cnri.reston.va.us/internet-drafts/draft-ylonen-ssh-protocol-00.txt). Also see “Ssh (Secure Shell) Remote Login Program” (http://www.cs.hut.fi/ssh/).

Stevens Stevens, W. “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”. IETF Draft. See ftp://ietf.cnri.reston.va.us/internet-drafts/draft-stevens-tcpca-spec-01.txt .

TerraVision A terrain visualization application that uses the DPSS to supply high-speed data streams for realtime, “virtual reality” type navigation of real landscape. (See http://www.ai.sri.com/~magic/terravision.html)

Tierney96 Tierney, B., W. Johnston, G. Hoo, J. Lee, “Performance Analysis in High-Speed Wide-Area ATM Networks: Top-to-Bottom End-to-End Monitoring”, IEEE Network, May, 1996, Vol. 10, no. 3. LBL Report 38246, 1996. (Also see http://www-itg.lbl.gov/DPSS/papers.html .)

Tierney94 “Distributed Parallel Data Storage Systems: A Scalable Approach to High Speed Image Servers”, B. Tierney, W. Johnston, L.T. Chen, H. Herzog, G. Hoo, G. Jin, J. Lee, and D. Rotem, Proceedings of ACM Multimedia `94, Oct. 1994, LBL-35408. Available as http://george.lbl.gov/ISS/papers/ISS-paper.ACM.final.html

Wiltzius Wiltzius, D., L. Berc, and S. Devadhar, “BAGNet: Experiences with an ATM metropolitan-area network”, ConneXions, Volume 10, No. 3, March 1996. Also see http://www.llnl.gov/bagnet/article.html .